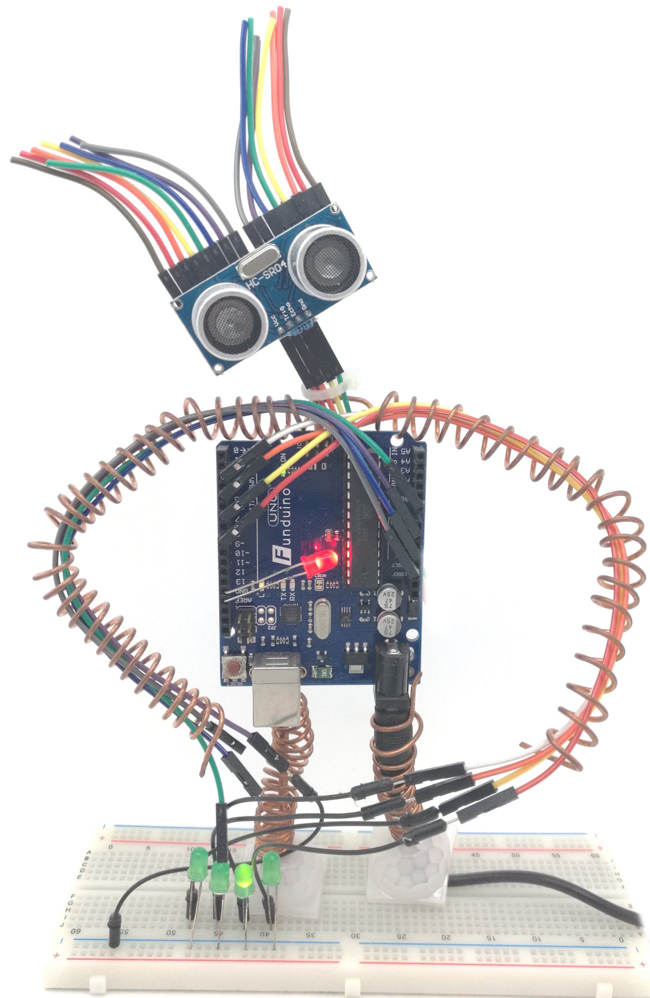
Arduino Mikrocontrolling



Inhaltsverzeichnis

1.Vorwort zur Arduino Anleitung

Diese Anleitung soll als Grundlage zum Erlernen der Arduino-Plattform dienen. Sie soll Anfängern einen einfachen, interessanten und eng geleiteten Einstieg in die Arduino-Thematik geben. Die Anleitung orientiert sich dabei hauptsächlich an praxisorientierten Aufgaben. Aber auch die theoretische Einführung sollte man vorab unbedingt lesen, um bei den späteren Praxisaufgaben nicht an fehlendem Grundwissen zu scheitern.

Was ist eigentlich „Arduino“?
Arduino ist eine Open-Source-Elektronik-Prototyping-Plattform für flexible, einfach zu bedienende Hardware und Software im Bereich Mikrocontrolling. Es ist geeignet, um in kurzer Zeit spektakuläre Projekte zu verwirklichen. Viele davon lassen sich unter dem Begriff „Arduino“ bei Youtube finden. Es wird vor allem von Künstlern, Designern, Tüftlern und Bastlern verwendet, um kreative Ideen zu verwirklichen. Aber auch in Schulen, Hochschulen und Universitäten wird die Arduino Entwicklungsumgebung zunehmend

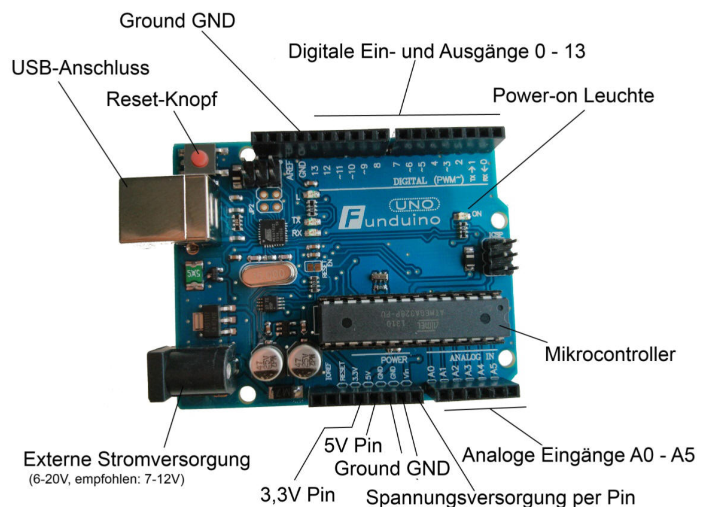
eingesetzt, um Lernenden einen kreativen und spannenden, aber vor allem auch einfachen Zugang zum Thema „Mikrocontrolling“ zu ermöglichen. Auch Themengebiete wie „Automatisierungstechnik“, „Robotik“ etc. lassen sich mit der Arduino Entwicklungsumgebung erarbeiten.

2. Hardware und Software

Der Begriff Arduino wird im allgemeinen Wortgebrauch gleichermaßen für die verschiedenen „Arduino-Boards“ (also die Hardware) als auch für die Programmierumgebung (Software) verwendet.

2.1 Hardware

Der „Arduino“ ist ein sogenanntes Mikrocontroller-Board (im weiteren Verlauf „Board“ genannt). Also im Grunde eine Leiterplatte (Board) mit jeder Menge Elektronik rund um den eigentlichen Mikrocontroller. Am Rand des Boards befinden sich viele Steckplätze (Pins genannt), an denen man die unterschiedlichsten Dinge anschließen kann. Dazu gehören: Schalter, LEDs, Ultraschallsensoren, Temperatursensoren, Drehregler, Displays, Motoren, Servos usw.



Es gibt sehr viele verschiedene Versionen von Mikrocontrollerboards, die mit der Arduino-Software verwendet werden können. Dazu gehören sowohl viele verschiedene große und kleine Boards mit der offiziellen „Arduino“ Bezeichnung als auch eine Vielzahl von häufig günstigeren Arduino-“compatiblen“ Boards. Die gängigsten Boards für die Arduino Entwicklungsumgebung haben neben der Markenbezeichnung wie „Arduino“ oder „Funduino“ die Namen „UNO“ und „MEGA 2560“.

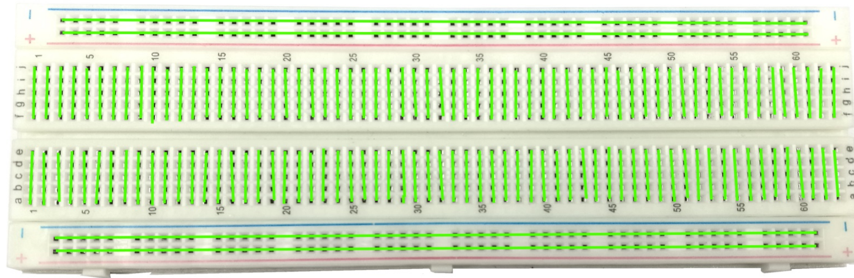
2.1.2 Empfohlenes Zubehör

Neben Sensoren und Aktoren benötigt man als Basis für schnelle und flexible Versuchsaufbauten Steckkabel in Verbindung mit einem Breadboard. Dadurch erspart man sich zeitraubende Lötarbeiten. Des Weiteren eignen sich Leuchtdioden sehr gut, um die Signalausgabe des Boards zu überprüfen.

2.1.2.1 Das Breadboard

Ein Breadboard oder auch „Steckbrett“ ist ein gutes Hilfsmittel, um Schaltungen aufzubauen ohne zu löten. In einem Breadboard sind immer mehrere Kontakte miteinander verbunden. Daher können an diesen Stellen viele Kabel miteinander verbunden werden, ohne dass sie verlötet oder verschraubt werden müssen.

Im folgenden Bild ist farbig dargestellt, welche Kontakte miteinander verbunden sind.



2.1.2.2 Leuchtdioden (LED)

Mit LEDs kann man sehr schnell die Ergebnisse eines Projekts testen. Daher sind sie für nahezu alle Arduino-Projekte nützlich. Über LEDs kann man vieles im Netz nachlesen. Hier nur die wichtigsten Infos.

• Der Strom kann nur in einer Richtung durch die LED fließen. Daher muss sie korrekt angeschlossen werden. Eine LED hat einen längeren und einen kürzeren Kontakt. Der längere Kontakt ist + und der kürzere ist -.

• Eine LED ist für eine bestimmte Stromstärke ausgelegt. Wird diese Stromstärke unterschritten, leuchtet die LED weniger hell oder sie bleibt aus. Wird die Stromstärke jedoch überschritten brennt die LED sehr schnell durch und wird an den Kontakten sehr heiß (ACHTUNG!). Eine zu hohe Stromstärke kann auftreten, wenn die für die LED maximale Spannung überschritten wird. Schließt man bspw. eine LED direkt an den 5V Ausgang an, wird sie umgehend kaputt sein. Daher verwendet man bei der Verwendung von LEDs an Arduinoboards immer einen Vorwiderstand.

• Typische Spannungswerte nach LED Farben: Blau:3,1V, Weiß:3,3V, Grün:3,7V, Gelb:2,2V, Rot:2,1V.

• Unverbindliche Empfehlung für Widerstände bei Verwendung der folgenden LED-Farben an den 5V Pins des Mikrocontrollers:

LED:	Weiß	Rot	Gelb	Grün	Blau	IR
Widerstand:	100 Ohm	200 Ohm	200 Ohm	100 Ohm	100 Ohm	100 Ohm

2.2 Software

Die Software, mit welcher der Mikrocontroller programmiert wird, ist open-Source-Software und kann auf www.arduino.cc kostenlos heruntergeladen werden. In dieser „Arduino-Software“ schreibt man dann kleine Programme, die der Mikrocontroller später ausführen soll. Diese kleinen Programme werden „Sketch“ genannt. Per USB-Kabel werden die fertigen Sketches dann auf den Mikrocontroller übertragen. Wie das funktioniert wird im Themengebiet „Programmieren“ behandelt.

2.2.1 Installation

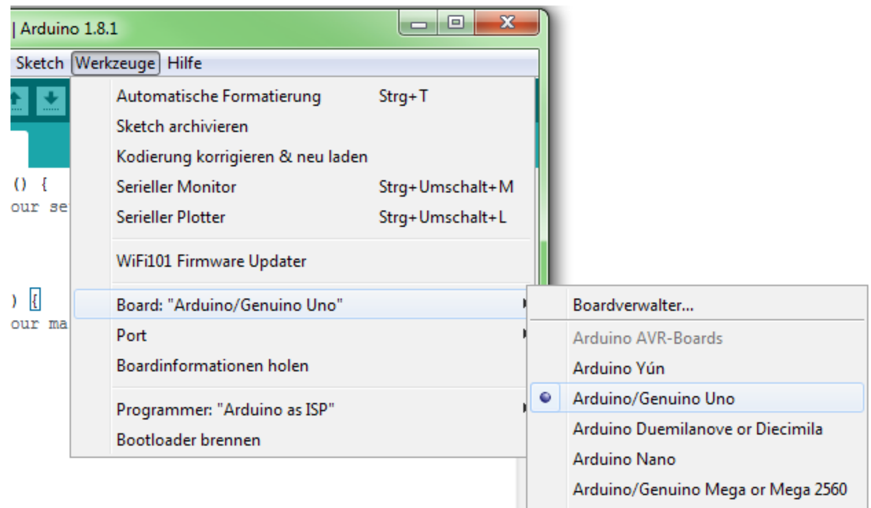
Nun muss nacheinander die Arduino-Software und der USB-Treiber für das Arduinoboard installiert werden.

2.2.1.1 Installation und Einstellung der Arduino-Software

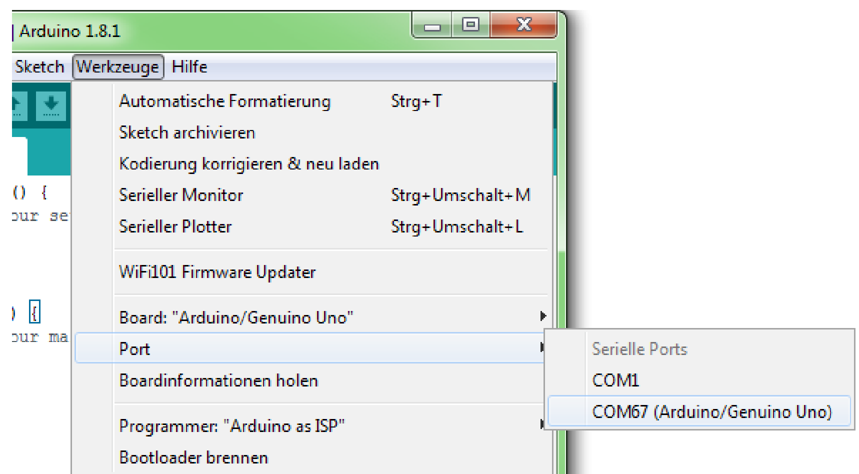
Die jeweils aktuellste Version der Arduinosoftware kann auf der Internetseite www.arduino.cc

heruntergeladen werden. Nach dem download beginnt die Installation, oder die Installation muss mit einem Doppelklick auf das heruntergeladene Programm gestartet werden. Während dieser Installation sollte noch kein Arduinoboard am Computer angeschlossen sein.
Nach der erfolgreichen Installation öffnet man den Softwareordner und startet das Programm mit der Datei arduino.exe.

Zwei wichtige Einstellungen gibt es im Programm zu beachten.
a) Es muss das richtige Board ausgewählt werden, das man am Computer anschließen möchte. Das „Funduino Uno“ Board wird hier als „Arduino Uno“ erkannt und das „Funduino MEGA2560“ Board entsprechend als „Arduino MEGA 2560“.



b) Es muss der richtige „Serial-Port“ ausgewählt werden. Das ist wichtig, damit der PC zuordnen kann, an welchem USB Anschluss das Board angeschlossen ist. Dies ist jedoch nur möglich, wenn der Treiber richtig installiert wurde. Das kann folgendermaßen geprüft werden:
Zum jetzigen Zeitpunkt ist der Arduino noch nicht am PC angeschlossen. Nun klickt man in dem Untermenü der Software auf „Serial Ports“. Dort werden schon ein oder mehrere Ports zu sehen sein (COM1 / COM4 / COM7 / …) Die Anzahl der angezeigten Ports ist dabei unabhängig von der Anzahl der USB-Anschlüsse. Wenn später das Board richtig installiert und angeschlossen ist, findet man hier einen weiteren Port.



2.2.1.2 Installation des USB-Treibers

Idealer Ablauf:

1. Mikrocontrollerboard an den PC anschließen
2. Der PC erkennt das Board und möchte einen Treiber installieren.

ACHTUNG: Der Treiber wird nicht immer automatisch erkannt und installiert. Man sollte in dem Fall im Verlauf der Installation den Treiber selber auswählen. Er befindet sich in dem Arduino-Programmordner in dem Unterordner „Drivers“.

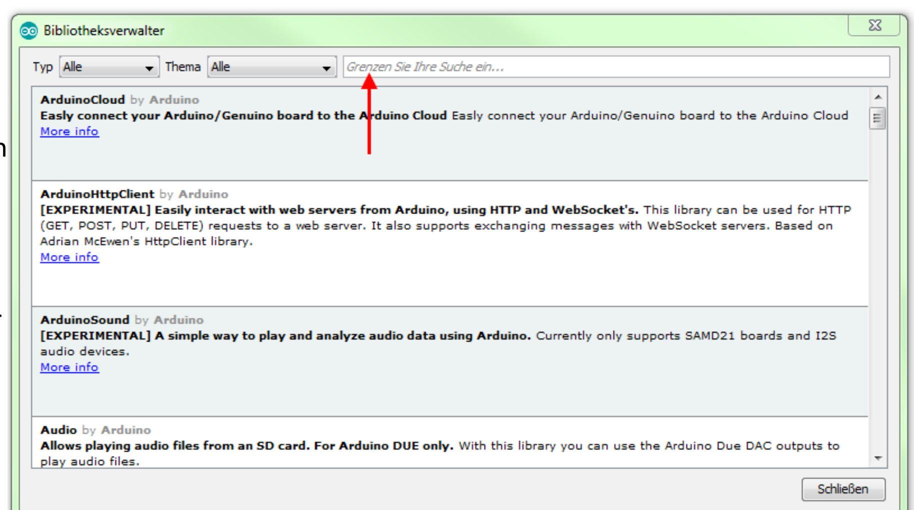
Kontrolle: In der Windows-Systemsteuerung des Computers findet man den „Gerätemanager“. Nach einer erfolgreichen Installation ist das Arduinoboard hier aufgelistet. Wenn die Installation nicht erfolgreich war, ist hier entweder nichts besonderes zu entdecken oder es ist ein unbekanntes USB-Gerät mit einem gelben Ausrufezeichen vorhanden. Für diesen Fall: Das unbekannte Gerät anklicken und auf „Treiber aktualisieren“ klicken. Jetzt kann man den Ablauf der manuellen Installation erneut durchführen

2.2.3 Bibliotheken zur Arduinosoftware hinzufügen

Eine Bibliothek (auch Library genannt) ist für einige Projekte sinnvoll, da diese die Programmierung vereinfachen kann. Es kann im Code auf Funktionen aus der Bibliothek zurückgegriffen werden, sodass diese nicht komplett im Sketch ausgeschrieben werden müssen. Im weiteren Verlauf der Anleitungen wird mehrfach auf solche Bibliotheken zurück gegriffen. Diese müssen erst in der Arduinosoftware hinzugefügt werden. Dazu gibt es verschiedene Möglichkeiten.

Die einfachste Möglichkeit bietet sich durch die Funktion „Bibliotheken verwalten…“. Diese befindet sich in der Software unter „Sketch > Bibliothek einbinden > Bibliotheken verwalten…“ Dort kann über das Suchfeld die gewünschte Library gesucht und direkt installiert werden.
Nach der erfolgreichen installation kann die Bibliothek direkt verwendet werden.
Mit der Installation von Programmbibliotheken werden häufig auch gleichzeitig Beispielsketche zur Arduinosoftware hinzugefügt. Diese Beispiele befinden sich unter „Datei > Beispiele“ und können einen guten Einblick in die einzelnen Funktionen der jeweiligen Bibliothek geben.

Es gibt auch die Möglichkeit eine Bibliothek auf einer externen Seite herunterzuladen und über die „.ZIP Bibliothek hinzufügen…“ Funktion einzubinden. Jedoch gestaltet sich diese Weise umständlicher als die zuvor beschriebene.



3. Programmieren

Damit ein Arduino Mikrocontroller das macht, was der Benutzer verlangt, wird mit Hilfe der Arduino-Software ein kleines Programm geschrieben. Dieses Programm wird in der Arduino Entwicklungsumgebung als „Sketch“ bezeichnet. Ein fertiger „Sketch“ wird nach der erfolgreichen Kontrolle in der Arduino-Software auf den Speicher des Mikrocontrollers geladen und dort unmittelbar ausgeführt.

3.1 Grundstruktur für einen Sketch

Ein Sketch kann zunächst in drei Bereiche eingeteilt werden.

1. Variablen benennen

Im ersten Bereich werden Elemente des Programms benannt. Zum Beispiel werden dort Variablen festgelegt oder sog. Programmbibliotheken geladen. Dieser Teil ist nicht für jeden Sketch zwingend erforderlich.

2. Setup

Das Setup wird vom Board nur einmal ausgeführt und ist zwingend erforderlich für jeden Sketch, selbst wenn in den Setupbereich keine Einträge erfolgen. Im Setup wird bspw. festgelegt, welcher Pin (Steckplatz für Kabel) am Mikrocontrollerboard ein Ausgang oder ein Eingang ist. Definiert als Ausgang kann an dem jeweiligen Pin eine Spannung ausgegeben werden (bspw. Um an diesem Pin eine LED leuchten zu lassen) und definiert als Eingang kann an dem Pin eine Spannung eingelesen werden (bspw. Die Spannungswerte eines Sensors).

3. Loop

Der Loop-Teil eines Sketches wird von Board kontinuierlich wiederholt. Daher kann dieser Teil auch als Hauptteil eines Sketches bezeichnet werden. Es verarbeitet den Sketch einmal komplett bis zum Ende und beginnt dann erneut am Anfang des Loop-Teils.

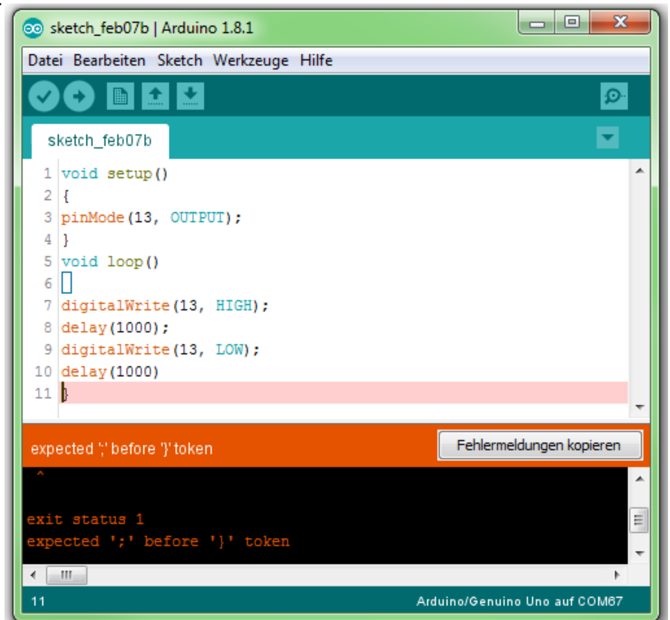
3.2 Häufige Fehlerquellen

Bei der Programmierung von Mikrocontrollern können sich an vielen Stellen Fehler einschleichen. Die häufigsten „Anfängerfehler“ während der Arbeit mit der Arduino-Software sind die folgenden beiden:

1) Das Board ist nicht richtig installiert oder es ist ein falsches Board ausgewählt. Beim hochladen des Sketches wird im unteren Bereich der Software eine Fehlermeldung angezeigt, die in etwa so aussieht wie rechts abgebildet. Im Fehlertext befindet sich dann ein Vermerkt „not in sync“. ->



2) Es gibt einen Fehler im Sketch. Beispielsweise ist ein Wort, eine Variable oder Befehl falsch geschrieben, oder es fehlt nur eine Klammer oder ein Semikolon. Im Beispiel links fehlt die geschweifte Klammer, die den Loop-Teil einleitet. Die Fehlermeldung beginnt dann häufig mit „expected…“. Das bedeutet, dass das Programm etwas erwartet, das noch nicht vorhanden ist.



3.3 Aufbau der Anleitungen

Die Struktur der folgenden Anleitungen ist jeweils sehr ähnlich.

1. Die Anleitungen beginnen mit einer Beschreibung der Aufgabe, die in der Anleitung abgearbeitet werden soll. Außerdem gibt es, falls erforderlich, eine Erklärung zu den verwendeten Bauteilen.
2. Vor dem jeweiligen Sketch befindet sich eine Skizze oder ein Foto um die Verkabelung aller Module zu verdeutlichen.
3. Die abgedruckten Sketche in den folgenden Anleitungen bestehen gleichzeitig aus Programmcode und einer Beschreibung, welche die Wirkung des jeweiligen Programmcodes erklärt.

Im **linken** Bereich ist der Programmcode in schwarzer oder farbiger Schrift abgedruckt, während sich um rechten Bereich hinter dem Zeichen „//“ in grauer Schrift die Erklärung zum Programmcode befindet. Die Erklärungen in grauer Schrift dürfen mit in in die Arduino-Software eingegeben werden und haben keinen Einfluss auf den Ablauf des Sketches.

Beispiel:

Links: Programmcode Rechts: Erklärungen zum Programmcode

```
void loop()                 //Hier beginnt das Hauptprogramm
{                           //Programmabschnitt beginnt.
digitalWrite(13, HIGH);     //Schalte die die Spannung an Pin13 ein (LED an).
delay(1000);                //Warte 1000 Millisekunden (eine Sekunde)
digitalWrite(13, LOW);      //Schalte die die Spannung an Pin13 aus (LED aus).
delay(1000);                //Warte 1000 Millisekunden (eine Sekunde).
}                           //Programmabschnitt beendet.
```

Wer sich nach diesem System durch die Programme arbeitet, wird den Programmcode in kurzer Zeit selber durchschauen und anwenden können. Danach kann man sich selbst mit weiteren Funktionen oder Modulen vertraut machen. Diese Anleitung stellt einen Einstieg in die Arduino Entwicklungsumgebung dar. Alle weiteren möglichen Programmfunktionen bzw. Programmcodes werden auf der Internetseite „www.arduino.cc“ unter dem Punkt „reference“ genannt.